NASA TECHNICAL MEMORANDUM

NASA TM X-53628

June 28, 1967

NASA TM X-53628

THE USE OF DSL / 90 FOR THE SIMULATION OF STABILIZED PLATFORM DYNAMICS

By R. A. LEWALLEN AND H. H. TRAUBOTH
Computation Laboratory

NASA

George C. Marshall Space Flight Center, Huntsville, Alabama

TECHNICAL MEMORANDUM X-53628

THE USE OF DSL / 90 FOR THE SIMULATION OF STABILIZED PLATFORM DYNAMICS

By

R. A. Lewallen*
and
H. H. Trauboth

George C. Marshall Space Flight Center
Huntsville, Alabama

ABSTRACT

To get a feel for the power of the digital simulation language DSL/90 the linearized X-loop of the stabilized platform for the Saturn guidance system was simulated by using this language. The computation times and relative errors at different step-sizes are measured and compared with each other.

NASA - GEORGE C. MARSHALL SPACE FLIGHT CENTER

* International Business Machines, Huntsville, Alabama

NASA - GEORGE C. MARSHALL SPACE FLIGHT CENTER

TECHNICAL MEMORANDUM X - 53628

THE USE OF DSL / 90 FOR THE SIMULATION OF STABILIZED PLATFORM DYNAMICS

By

R. A. Lewallen
and
H. H. Trauboth

COMPUTATION LABORATORY
RESEARCH AND DEVELOPMENT OPERATIONS

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

TECHNICAL MEMORANDUM X- 53628

THE USE OF DSL / 90 FOR THE SIMULATION OF STABILIZED PLATFORM DYNAMICS

SUMMARY

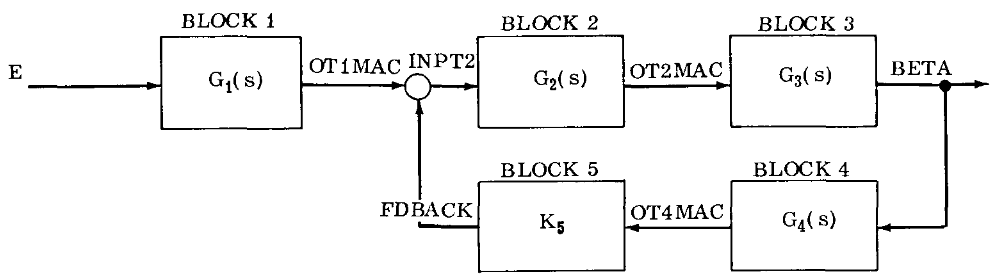
To get a feel for the power of the digital simulation language DSL/90 the linearized X-loop of the stabilized platform for the Saturn guidance system was simulated by using this language. The computation times and relative errors at different step-sizes are measured and compared with each other.

INTRODUCTION

The rapid development of computer technology and the creation of new engineering oriented languages have obviously established that general purpose digital computers are increasingly suitable for simulation of the dynamics of large physical systems. At Marshall Space Flight Center (MSFC), Huntsville, Alabama, a combined effort of the Computation Laboratory and Quality and Reliability Assurance Laboratory has been undertaken to simulate continuous and discrete dynamics of an aerospace vehicle and its ground support equipment (GSE) on a large digital computer. This Aerospace Vehicle Simulation (AVS), also called Launch Vehicle Component Level Simulation (LVCLS), is primarily being developed as a major tool for the checkout of space vehicles; but hopefully it will become an essential part of an integrated engineering information system which can be used by various laboratories at MSFC for the design, checkout, and management of space vehicles.

In order to facilitate the input of data into the digital computer and have an engineer rather than a programmer control the simulation, several simulation languages have been developed. One of the most promising simulation languages is DSL/90 [1-2]. This language allows an engineer to write a program in close resemblance to the block diagram which describes mathematically the continuous dynamics of the system to be simulated. DSL/90 also allows one to construct any functional block as a macro using FORTRAN statements and to unite macro statements with regular FORTRAN statements. The user has the choice of six standard integration methods to obtain maximum computation speed. However, he is free to design his own integration scheme and call it up whenever he needs it.

For a large scale digital simulation such as AVS, it is necessary to make use of a simulation language. In order to get a feeling for the computation time of DSL/90 and to compare it with that of other available languages, it was proposed to run simulations of a test case using all available integration schemes. The linearized X-loop of the stabilized platform for the Saturn V guidance system was chosen as the test case. The block diagram of this control system, which consists of four rather complex transfer function blocks, is shown in Figure 1. The total system contains seven real and four complex poles and eight real and one complex zeros.



TRANSFER FUNCTIONS:

$$G_1(s) = \frac{0.00724115\left(1+\frac{s}{17.8069}\right)\left(1+\frac{s}{138.318}\right)\left(1+\frac{s}{187.192}\right)s}{\left(1+\frac{s}{1.5662}\right)\left(1+\frac{s}{16.2565}\right)\left(1+\frac{s}{136.889}\right)\left(1+\frac{2\times 0.377453\ S}{531.122}+\frac{s^2}{531.122^2}\right)}$$

$$G_2(s) = \frac{1+\frac{s}{5499}}{\frac{s^2}{1759^2}+\frac{2\times 0.34\ S}{1759}+1}$$

$$G_3(s) = \frac{\frac{9.85387\times 10^4}{2040.82}}{s\left(\frac{s^2}{39^2}+1\right)}$$

$$G_4(s) = \frac{0.0941349\left(1+\frac{s}{16.2512}\right)\left(1+\frac{s}{17.8056}\right)\left(1+\frac{s}{187.192}\right)\left(1+\frac{2\times 0.498219\ S}{41.3356}+\frac{s^2}{41.3356^2}\right)}{\left(1+\frac{s}{1.5662}\right)\left(1+\frac{s}{16.2565}\right)\left(1+\frac{s}{136.889}\right)\left(1+\frac{2\times 0.377453}{531.122}\ s\times\frac{s^2}{(531.122)^2}\right)}$$

$K_5 = 572.958$

Input signal E = 10 volts step function at t = 0

FIGURE 1. BLOCK DIAGRAM OF LINEARIZED $\dot{X}$-LOOP OF STABILIZED PLATFORM

POLYNOMIAL TRANSFER FUNCTION GENERATOR

A preliminary analysis of the block diagram indicated that the standard "blocks" present in DSL/90 could not handle conveniently all the transfer functions in the control system. Therefore, it was decided to use the General Transfer Function Macro (GTFM-BLOCK) [3].

Since this macro accepts as input the coefficients of a transfer function in polynomial form, a FORTRAN program was written to automate the conversion process from the factored form (zeros and poles) to the polynomial form of the transfer functions.

The FORTRAN program is described in Appendix A.

DSL / 90 - SIMULATION PROGRAM

There are two ways of handling transfer function blocks. Either one uses the three standard DSL/90 blocks [1] or one applies the General Transfer Function Macro (GTFM) Block, a copy of which was obtained from IBM Federal Systems Division, Huntsville. Appendix B explains the GTFM-Block in detail. Figure 2 shows the breakdown of the various blocks into standard blocks and GTFM blocks.

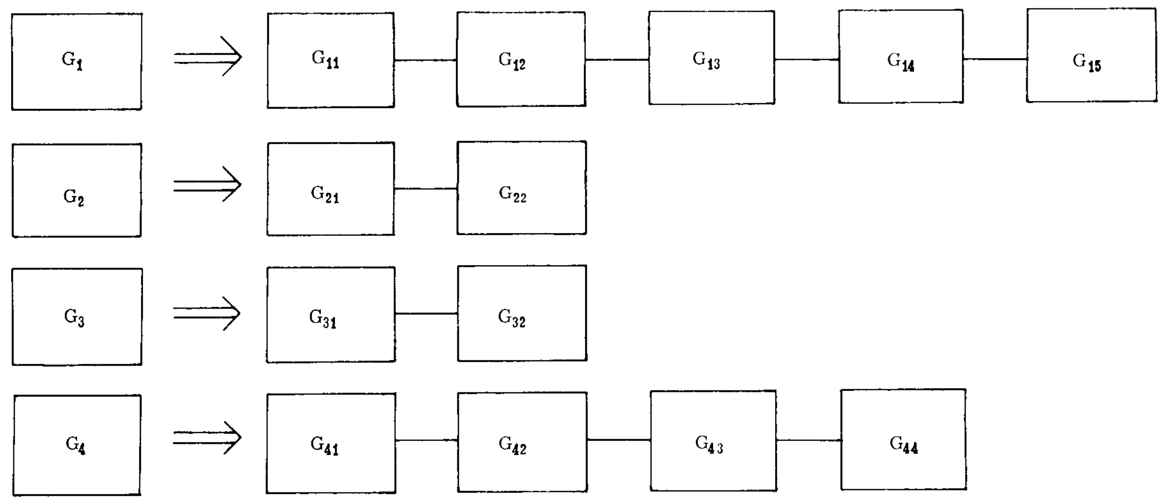
Initial runs of the complete system indicated that input errors were creating an unstable condition. In order to isolate these errors and correct them, each block in the system was run individually with a step input using both the macro method and the standard DSL/90 blocks.

Several simulation runs with the following six standard integration methods have been made to compare the computation times.

1. Milne fifth-order predictor-corrector integration routine, variable step-size (MILNE);

2. Runge-Kutta fourth-order, variable step size (RKS);

3. Runge-Kutta fourth-order, fixed step size (RKSFX);



$$G_{11} = \frac{1 + \frac{s}{17.8069}}{1 + \frac{s}{1.5662}} \; ; \qquad G_{12} = \frac{1 + \frac{s}{138.318}}{1 + \frac{s}{16.2565}} \; ; \qquad G_{13} = \frac{1 + \frac{s}{187.192}}{1 + \frac{s}{136.889}} \; ; \qquad G_{14} = \frac{1}{1 + \frac{2 \times 0.377453\, s}{531.122} + \frac{s^2}{531.122^2}} \; ; \qquad G_{15} = 0.00724115\, \frac{1 + 1000 s}{1 + \frac{s}{1000}} \; ;$$

$$G_{21} = \frac{1}{1 + \frac{2 \times 0.34\, s}{1759} + \frac{s^2}{1759^2}} \; ; \qquad G_{22} = \frac{1 + \frac{s}{5499}}{1 + \frac{S}{1000}} \; ;$$

$$G_{31} = \frac{1}{s} \; ; \qquad G_{32} = \frac{\frac{9.85387 \times 10^4}{2040.82}}{1 + \frac{s^2}{39^2}} \; ;$$

$$G_{41} = \frac{1 + \frac{s}{16.2512}}{1 + \frac{s}{1.5662}} \; ; \qquad G_{42} = \frac{1 + \frac{s}{17.8056}}{1 + \frac{s}{16.2565}} \; ; \qquad G_{43} = \frac{1 + \frac{s}{187.192}}{1 + \frac{s}{136.889}} \; ; \qquad G_{44} = 0.0941349\, \frac{1 + \frac{2 \times 0.498219\, s}{41.3356} + \frac{s^2}{41.3356^2}}{1 + \frac{2 \times 0.377453\, s}{531.122} + \frac{s^2}{531.122^2}} \; .$$

FIGURE 2. BREAKDOWN OF BLOCKS OF BLOCK DIAGRAM IN FIGURE 1 INTO DSL/90-STANDARD AND GTFM BLOCKS

4. Simpson Rule, fixed step size (SIMP);

5. Trapezoidal Rule, fixed step size (TRAPZ);

6. Rectangular Rule, fixed step size (RECT).

The relative error criterion was set to 10^{-5}, 10^{-4} and 10^{-2} for the variable step-size methods. For the fixed step-size methods the step size was adjusted in relation to the minimum step size of Runge-Kutta (variable step size), i.e., to $\Delta t = 2.5 \times 10^{-5}$ and 1×10^{-4} sec, respectively.

There are three phases of operations:

1. Translation of DSL/90 program into FORTRAN;

2. Compilation of FORTRAN into machine language;

3. Execution of machine language program.

If the integration method is changed no retranslation and recompilation are necessary. The binary deck for executing the simulation run is the same except one control card defining the integration method has to be changed. If a rerun with different parameters is to be made, the affected parameter cards only have to be changed without a retranslation and recompilation being necessary.

The DSL/90 program "Simulation of Stabilized Platform" is shown in Appendix C.

RESULTS OF SIMULATION

The input to the stabilized platform is a step function of E = 10 volts. Of the six output signals the output signals BETA, FDBACK and OT1MAC are chosen to be plotted for the real-time t = 0 to t = 0.2 sec (Fig. 3). A print-out is given in Figure 4. Figure 5 shows the characteristics of the individual blocks 2, 3 and 4, i.e., their responses to a step-function.

Various computation times for the six integration methods available in DSL/90 have been measured for different step-sizes and error criteria. They are tabulated in Tables I through III. All runs were made on the IBM 7094-II with 90 kc tape drives located at the Computation Laboratory.

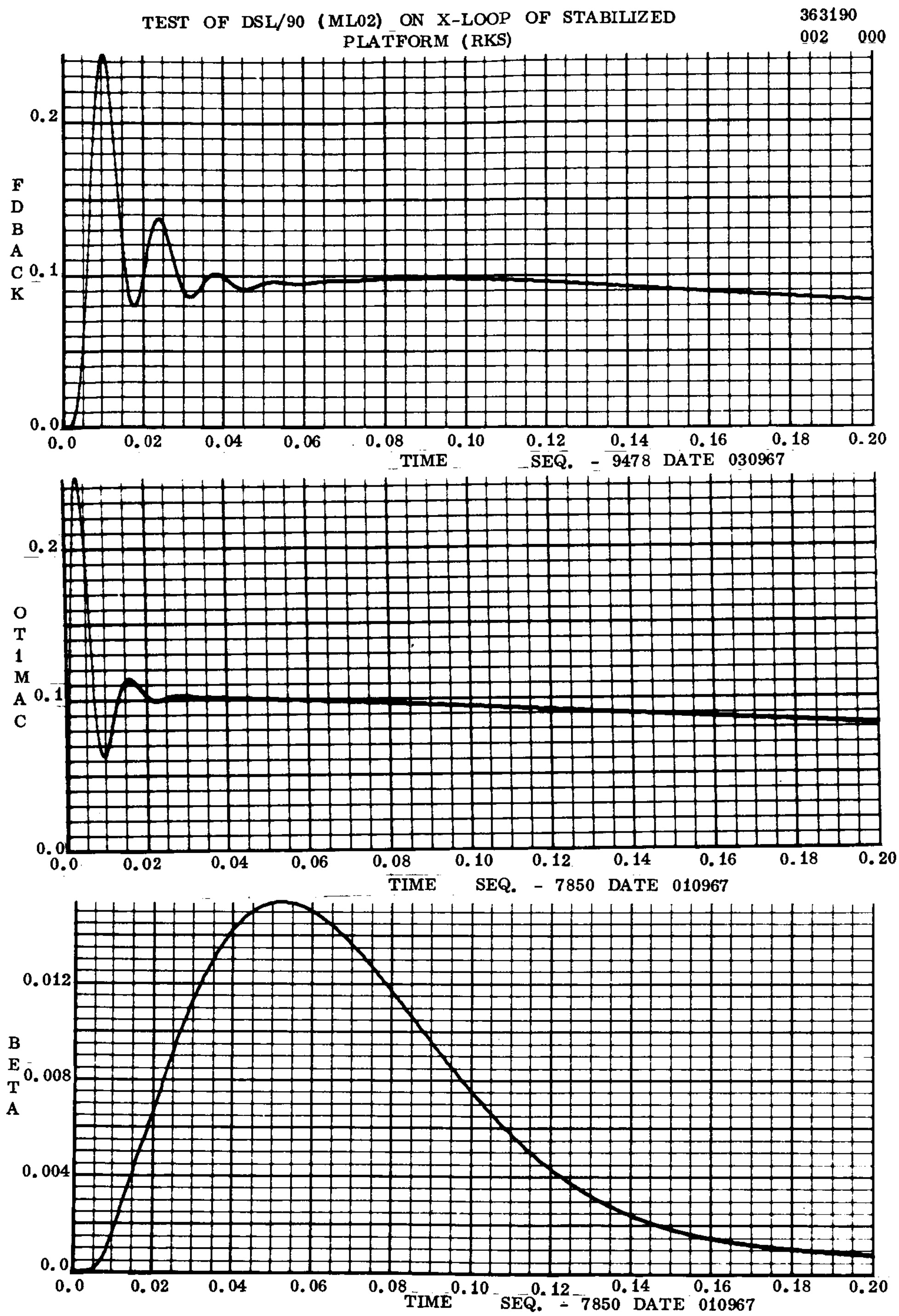


FIGURE 3. OUTPUT SIGNALS FDBACK, OT1MAC AND BETA AS RESPONSES TO INPUT SIGNAL E, WHICH IS A STEP FUNCTION

TEST OF DSL/90 (ML02) ON X-LOOP OF STABILIZED PLATFORM (RKSFX)

TIME	OT1MAC	INPT2	OT2MAC	BETA	OT4MAC	FDBACK	DELT
0.000E-39	0.0000E-39	0.0000E-39	-0.0000E-39	0.0000E-39	-0.0000E-39	-0.0000E-39	2.5000E-05
2.000E-02	1.0260E-01	2.5168E-03	6.8321E-03	6.6815E-03	1.7468E-04	1.0008E-01	2.5000E-05
4.000E-02	1.0157E-01	2.7317E-03	2.1320E-03	1.4244E-02	1.7251E-04	9.8839E-02	2.5000E-05
6.000E-02	9.9713E-02	5.1580E-03	5.2599E-03	1.5013E-02	1.6503E-04	9.4555E-02	2.5000E-05
8.000E-02	9.7590E-02	1.0168E-04	1.3548E-04	1.1694E-02	1.7015E-04	9.7489E-02	2.5000E-05
1.000E-01	9.5265E-02	-2.4441E-03	-2.4310E-03	7.5333E-03	1.7053E-04	9.7709E-02	2.5000E-05
1.200E-01	9.2823E-02	-2.9266E-03	-2.9302E-03	4.3206E-03	1.6711E-04	9.5749E-02	2.5000E-05
1.400E-01	9.0319E-02	-2.2382E-03	-2.2476E-03	2.4114E-03	1.6154E-04	9.2557E-02	2.5000E-05
1.600E-01	8.7792E-02	-1.3328E-03	-1.3409E-03	1.4662E-03	1.5555E-04	8.9125E-02	2.5000E-05
1.800E-01	8.5272E-02	-6.7005E-04	-6.7495E-04	1.0308E-03	1.5000E-04	8.5942E-02	2.5000E-05
2.000E-01	8.2777E-02	-3.1859E-04	-3.2060E-04	7.9423E-04	1.4503E-04	8.3096E-02	2.5000E-05

VARIABLE	MINIMUM	TIME	MAXIMUM	TIME
OT1MAC	0.0000E-39	0.0000E-39	2.4662E-01	3.1500E-03
INPT2	-1.7896E-01	1.0025E-02	2.3798E-01	2.8750E-03
OT2MAC	-1.8717E-01	1.0175E-02	2.7044E-01	2.9500E-03
BETA	0.0000E-39	0.0000E-39	1.5401E-02	5.2200E-02
OT4MAC	-0.0000E-39	0.0000E-39	4.2627E-04	1.0275E-02
FDBACK	-0.0000E-39	0.0000E-39	2.4424E-01	1.0275E-02
DELT	2.5000E-05	0.0000E-39	2.5000E-05	0.0000E-39

DSL/90 SIMULATION TIME = 46.883 SECONDS

FIGURE 4. PRINTOUT OF SIMULATION RUN OF BLOCK DIAGRAM IN FIGURE 1 USING RUNGE-KUTTA INTEGRATION WITH FIXED STEP SIZE

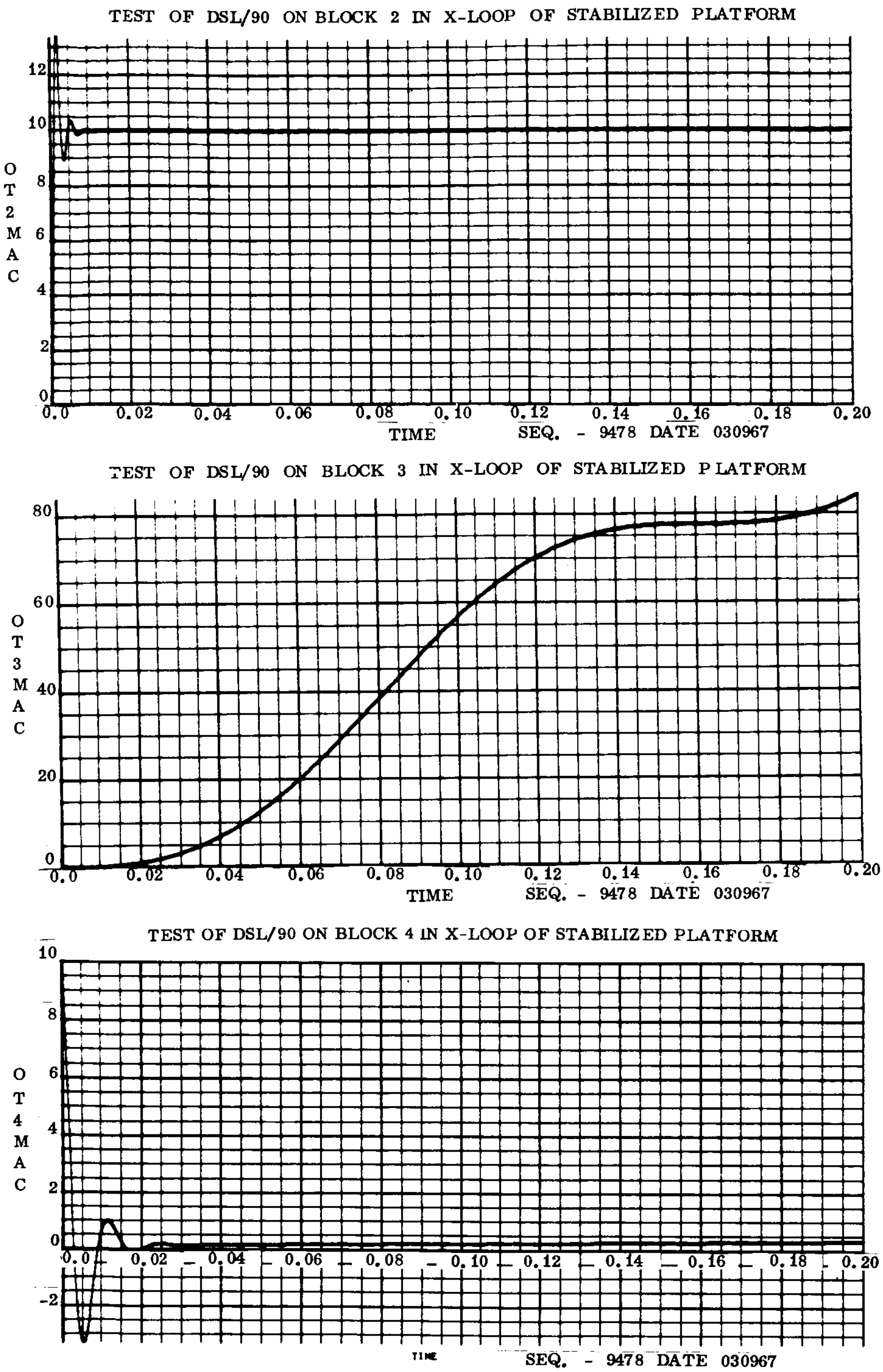


FIGURE 5. STEP FUNCTION RESPONSES OF BLOCK 2, BLOCK 3 AND BLOCK 4

TABLE I. TIMES FOR PREPARATION OF SIMULATION

Read control cards, mount and rewind tapes	27 sec x)
Load translator portion of DSL/90	19 sec
Translate DSL/90 program to FORTRAN	12 sec
Compile FORTRAN	17 sec
Load simulator portion of DSL/90	23 sec x)
Total	98 sec

x) These times must also be taken into account, if no compilation and translation precede the execution.

TABLE II. COMPARISON OF ERRORS AND RUN TIMES FOR VARIABLE STEP METHODS

Method	Error Criterion RELERR	Δt_{min} in sec	Δt_{max} in sec	Percent Error based upon final values			Percent Error based upon maximum values			Execution Time in sec
				BETA	OT1MAC	FDBACK	BETA	OT1MAC	FDBACK	
Milne	10^{-5}	7×10^{-4}	6.25×10^{-4}	0.01	<0.005	<0.005	< 2.52	58.40	0.98	11.7
Milne	10^{-4}	1.6×10^{-4}	2×10^{-3}	0.005	<0.005	<0.005	2.52	58.40	59.02	6.5
Milne	10^{-2}	3.1×10^{-4}	2×10^{-3}	0.005	<0.005	<0.005	2.52	58.42	59.02	8.9
Runge-Kutta (Variable)	10^{-5}	1×10^{-4}	1.63×10^{-3}	<0.005	<0.005	<0.005	<0.005	0.02	<0.005	7.1
Runge-Kutta (Variable)	10^{-4}	1.25×10^{-4}	4.1×10^{-3}	<0.005	<0.005	<0.005	0.04	0.42	0.09	2.6
Runge-Kutta (Variable)	10^{-2}	5×10^{-4}	4.3×10^{-3}	<0.005	<0.005	<0.005	0.01	0.98	0.19	2.0

TABLE III. COMPARISON OF ERRORS AND RUN TIMES FOR FIXED STEP METHODS

Method	Δt in sec	Percent Error based upon final values			Percent Error based upon maximum values			Execution Time in sec	
		BETA	OT1MAC	FDBACK	BETA	OT1MAC	FDBACK		
Runge-Kutta (Fixed)	2.5×10^{-5}							46.9	
Runge-Kutta (Fixed)	1×10^{-4}	0.01	<0.005	<0.005	<0.005	0.02	<0.005	12.8	
Simpson	2.5×10^{-5}	<0.005	<0.005	<0.005	<0.005	<0.005	<0.005	31.4	
Simpson	1×10^{-4}	0.01	<0.005	<0.005	<0.005	0.04	0.01	7.5	
Trapezoidal	2.5×10^{-5}	<0.005	<0.005	<0.005	<0.005	<0.005	<0.005	33.8	
Trapezoidal	1×10^{-4}	0.01	<0.005	<0.005	<0.005	0.05	0.01	8.6	
Rectangular	2.5×10^{-5}	0.04	<0.005	<0.005	0.06	0.65	1.00	11.8	
Rectangular	1×10^{-4}	0.14	<0.005	<0.005	0.23	2.67	4.20	3.1	

The total time to prepare the simulation, which includes translation of the DSL/90 program to FORTRAN and the compilation of the FORTRAN-program, amounts to about 100 sec. This time can be cut in half for a DSL/90 program which has already been compiled. The results of Runge-Kutta Fixed Step Size with $\Delta t = 2.5 \times 10^{-5}$ sec are considered as the most accurate values and, therefore, are used as reference.

Table II shows that the Milne method did not lead to a usable solution for this system with any of the relative errors specified. This seems to be caused by the low order (rectangular) starter used in the present system and to the tendency of the Milne method to become unstable if the system contains high-frequency terms [5, 6]. However, satisfactory results have been obtained with the Milne method by decreasing the maximum step-size (specifying a small DELMAX) at the expense of longer execution time.

If only signal BETA is requested, a larger step-size can be tolerated, resulting in decreased execution time (e.g., 0.1 percent accuracy in 2.6 sec for Runge-Kutta Variable Step). However, if other signals, such as OT1MAC, are requested, the step-size has to be decreased considerably to obtain the same accuracy as for BETA. This is caused by the filtering or smoothing effect of Block 3 (Fig. 5). For this example, Runge-Kutta Variable Step gave the most favorable results with respect to both accuracy and computation time.

CONCLUSIONS

The total time to perform a simulation after the mathematical model of the physical system has been established, consists of the time for the following procedures.

1. Writing the DSL/90 program and punching it into cards;

2. Initializing the DSL/90 processor, translating the DSL/90 program into a FORTRAN program and compiling the FORTRAN program;

3. Executing the compiled program.

The execution time of about 3 sec is relatively short for the simulation of a medium-size control system with an engineering accuracy of about 0.1 percent. The simulation time is sufficient for on-line non-real-time applications, though it is by a factor of about 15 too slow for real-time applications.

The preparation time of about 100 sec for the punched DSL/90 program is nearly thirty times the execution time but still shorter than the time required to put a patch board on an analog computer and to adjust its potentionmeters. If the same system is to be simulated with different parameters, the preparation time shrinks to only 50 sec.

Writing a DSL/90 program requires less programming knowledge than writing a FORTRAN program; however, it still requires some programming experience and will not always be readily handled by a practical engineer. Nevertheless, it is easier to set up a DSL/90 program than an analog program since no scaling, no conversion of transfer functions into differential equations, and no hardware limitations are encountered.

As long as the digital simulation is operated in a batch process mode via punched cards, the engineer may prefer the analog computer since he has closer communications with the machine. Therefore, any digital simulation language such as DSL/90 will gain if it is used on-line, preferably with computer graphics.

Though the computation times and numerical errors of the different integration methods cannot be generalized, they give a feeling for the order of magnitude one can expect. In particular, although our results using the Milne predictor-corrector were unfavorable, other users have reported very good results when using Milne's method for problems with somewhat different characteristics.

APPENDIX A

POLYNOMIAL TRANSFER FUNCTION GENERATOR*
FORTRAN IV

Purpose and Method

The purpose here is to generate a transfer function polynomial, given a factored transfer function. We use a transfer function of the form:

$$k(1 + s/p_1)\ (1 + s/p_2)\ \ldots\ (1 + s/p_m)$$

or

$$k(1 + s/p_1)\ (1 + s/p_2)\ \ldots\ (1 + s/p_{m-2})\ \left(1 + \frac{2 \cdot p_{m-1}}{p_m} s + \frac{s^2}{p_m^2}\right).$$

This function is input $(k, p_1, p_2, \ldots, p_m)$, yielding the equation $c_0 + c_1 s + c_2 s^2 + \ldots + c_m s^m$ in the form $(c_0, c_1, \ldots, c_m)$. The p_i's are converted to double precision complex numbers and subroutine GENER is called to generate the polynomial.

Usage

Load the binary deck, followed by the data cards (see Fig. A-1).

FØRMAT (I3): NØCASE - number of polynomials to be generated.

FØRMAT (I3): m - number of coefficients in this polynomial.
(If quadratic term appears, replace m by -m.)

* Appendix was prepared by R. A. Lewallen, IBM Corporation, Huntsville, Alabama, and Alex Zakson, General Electric Company, Huntsville, Alabama.

FØRMAT (6E12.6) k - constant multiplier of expression.

FØRMAT (6E12.6) p_i's - see METHOD for example of these.
p_i's are input exactly as they appear here.

Repeat m, k, and p_i's NØCASE times.

ACCURACY: Output appears in double precision. Up to 8 positions may be accurate, depending on accuracy of input.

RESTRICTIONS: $m \leq 99$

```
$IBJOB GO
$IBFTC MAIN      LIST,DECK
      DIMENSION  P(99)
      DOUBLE PRECISION  Q(199)
 1    FORMAT  (I3)
 2    FORMAT  (6E12.6)
 3    FORMAT  (//14H  INPUT  DATA  IS/8X,  1HC,  13X, 7HP-S  ARE/(6E20.8) )
 4    FORMAT  (40H  COEFFICIENTS APPEAR IN DESCENDING ORDER)
C     NOCASE IS NUMBER OF POLYNOMIALS DESIRED.
      READ (5,1)  NOCASE
      DO  500 J = 1, NOCASE
C     IF NO QUADRATIC TERM APPEARS,  LET  M  =  M.
C     IF EQUATION INCLUDES QUADRATIC TERM,  LET M  =  -M.
      READ  (5,1)  M
      NOTERM  =  M
      M  =  IABS( M)
      READ  (5,2)  C,  (P(I) ,   I = 1,M)
      WRITE  (6,3)  C,  (P(I) ,  I = 1,M)
      WRITE (6,4)
      N = M
C     DOES QUADRATIC TERM APPEAR.
      IF (NOTERM  .LT. 0)  N  =  M-2
      IF  (N  . LE.  0)  GO TO 110
      DO  100  I  =  1,N
      Q( 2*I -1)  =  -P( I)
      Q( 2*I)  =  0.
      C  =  C/P( I)
 100  CONTINUE
 110  IF  (NOTERM  .GT.  0)  GO  TO  190
      P( M-1)  =  2.  *  P( M-1)  /  P( M)
      P( M)  =  1.  /  P( M)  ** 2
      C  =  C  *  P( M)
C     COMPUTE  ROOTS  OF  QUADRATIC.
      DISCR  =  P( M-1)  **  2  -  4. *P( M)
      IF  (DISCR)  25,  75,  75
 25   Q( 2 * M-3)  =  - P( M-1) /  (2.  *P( M)  )
      Q( 2 *M-1)  =  Q( 2 *M -3)
      Q( 2 *M -2)  =  SQRT  (ABS  (DISCR)  )  /  (2.  *P( M)  )
      Q( 2 *M)  =  -Q( 2 * M-2)
      GO  TO  190
 75   ADD  =  SQRT  (DISCR)  /  (2. *P(M)  )
      START  =  -P( M-1)  /  (2.  *  P( M)  )
      Q( 2 *M - 3)  =  START  +  ADD
      Q( 2 *M - 1)  =  START  -  ADD
      Q( 2 *M - 2)  =  0.
      Q( 2 *M)  =  0.
 190  Q( 2 *M +1)  =  C
 200  CALL GENER  (Q,M)
 500  CONTINUE
      STOP
      END
```

FIGURE A1. POLYNOMIAL TRANSFER FUNCTION GENERATOR

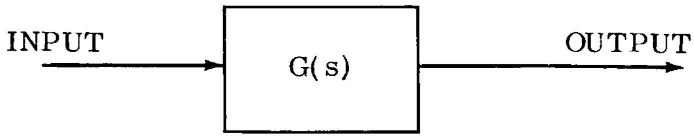
APPENDIX B

GENERAL TRANSFER FUNCTION MACRO BLOCK

Purpose

This appendix contains a description of a macro that extends DSL/90 block capability to general transfer functions.

For the general transfer function of the form:



$$G(s) = \frac{b_{n+1}\, S^{n} + b_{n}\, S^{n-1} + \ldots + b_2 S + b_1}{a_{n+1}\, S^{n} + a_{n}\, S^{n-1} + \ldots + a_2 S + a_1}$$

where:

$$a_{n+1} \neq 0$$

$$1 \leq n \leq K\ .$$

A block diagram method described in detail in Reference 4 can be used to reduce this transfer function to block notation. The block notation for this transfer function is shown in Figure B1. This block diagram can be incorporated easily into a DSL/90 macro.

Description

For user convenience, a simplified macro argument and table are used in this macro generation. Also for computer run time minimization, three values of K = 5, 10, 30 have been used. These macros are called T5FCN, T10FCN, and T30FCN, respectively.

To utilize this macro efficiently, the user must supply the numerator and denominator coefficients in the form of table arrays. The user simply picks the proper macro range (K = 5, 10, 30) and orders the coefficient arrays. In particular, for the following m^{th} numerator polynomial:

$$N_{m+1} S^m + N_m S^{m-1} + \ldots + N_2 S + N_1$$

the order of the coefficients for the macro would be:

$$b_1, b_2, b_3, \ldots b_{K-m}, b_{K+1-m}, \ldots b_{K-1}, b_K, b_{K+1}$$

where: (a) The first (K - m) coefficients are loaded as zero. That is:

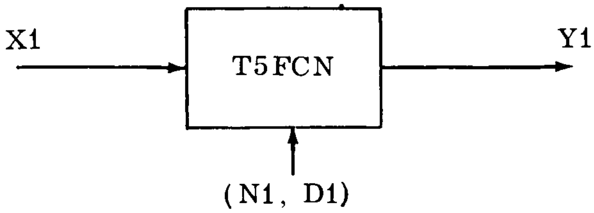
$$b_1 = b_2 = \ldots = b_{k-m-1} = b_{K-m} = 0$$

(b) The last (m + 1) coefficients are the specific m^{th} degree transfer coefficients <u>in increasing order as power of S</u>. That is for this case:

$$\underbrace{0, 0, \ldots 0,}_{K-m} \underbrace{N_1, N_2, \ldots N_m, N_{m+1}}_{m+1}$$

The order of the denominator follows the numerator order exactly. It is to be emphasized that this method assumes the degree of the numerator is equal to the degree of the denominator. If the numerator degree is less than the denominator it may be manipulated by using appropriate numerator coefficients as zero.

The macro is specified by calling it as:



Y1 = T5FCN (N1, D1, X1)

where: (1) N1 is a numerator array of (5 + 1) elements,
(2) D1 is a denominator array (5 + 1) elements, and
(3) X1 is the single input to this macro block.

The numerator and denominator arrays are specified in a table as:

TABLE N1 (1 - 6) = b_1, b_2, b_3, b_4, b_5, b_6, . . .

D1 (1 - 6) = a_1, a_2, a_3, a_4, a_5, a_6 .

The macros for 10 and 30 degree polynomials have the same format as T5FCN. The use of this macro is illustrated in Appendix 3.

$$G(s) = \frac{\text{OUTPUT}}{\text{INPUT}} = \frac{b_{n+1}S^n + b_nS^{n-1} + \ldots + b_2S + b_1}{a_{n+1}S^n + a_nS^{n-1} + \ldots + a_2S + a_1}$$

$a_{n+1} \neq 0$

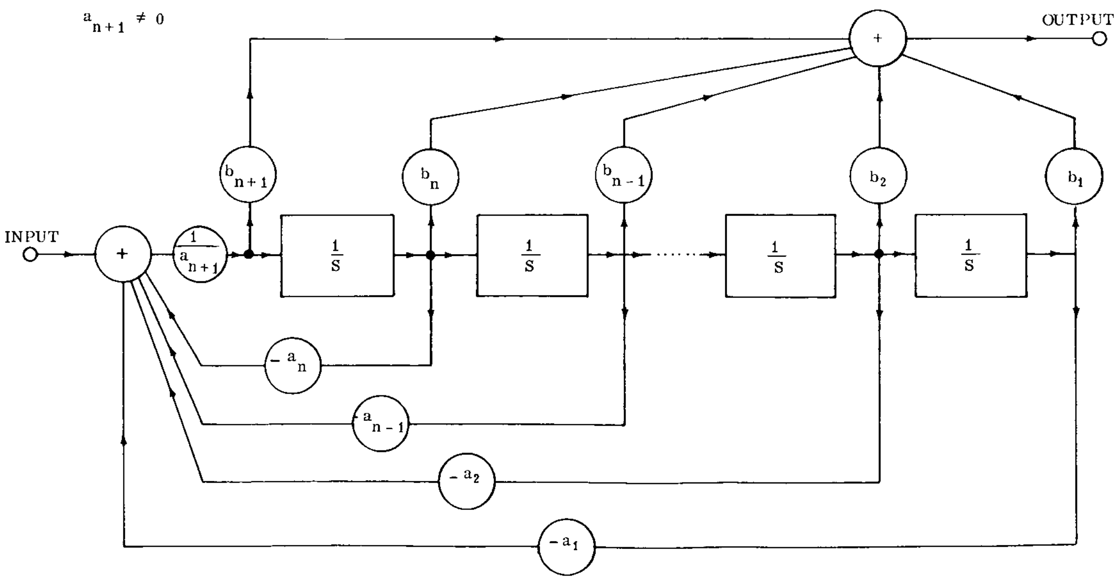


FIGURE B1. BLOCK DIAGRAM FOR GENERAL TRANSFER FUNCTION

APPENDIX C

DSL / 90 - PROGRAM "SIMULATION OF STABILIZED PLATFORM"

```
$IEDIT          SYSCK1,SRCH
$IBLDR MAIN
$IBLDR CENTRL
$IBLDR SEQNM
$IEDIT
DECK
```

MACRO DEFINITION

```
TITLE           TEST OF DSL/90 ON X-LOOP OF STABILIZED PLATFORM -- MOD LEVEL 2
MACRO OUT =     T5FCN(N,D,IN)
                M1 = INTGRL(0.0,M)
                M2 = INTGRL(0.0,M1)
                M3 = INTGRL(0.0,M2)
                M4 = INTGRL(0.0,M3)
                M5 = INTGRL(0.0,M4)
                M = (1.0/D(6)) * (IN - D(5) * M1 - D(4) * M2 - D(3) * M3 - D(2) * M4 - D(1) * M5)
                OUT = N(6) * M + N(5) * M1 + N(4) * M2 + N(3) * M3 + N(2) * M4 + N(1) * M5
ENDMAC
```

INTERCONNECTION STATEMENTS

```
STORAG NUMCF1(6), DENCF1(6), NUMCF2(6), DENCF2(6),                         ...
                NUMCF4(6), DENCF4(6)
***             COMPUTE BLOCK 1 WITH USER-SUPPLIED MACRO         ***
                STEPIN = 10.0 * STEP(0.0)
                OT1MAC = T5FCN (NUMCF1, DENCF1, STEPIN)
***             COMPUTE BLOCK 2 WITH USER-SUPPLIED MACRO         ***
                INPT2 = OT1MAC - FDBACK
                OT2MAC = T5FCN (NUMCF2, DENCF2, INPT2)
***             COMPUTE BLOCK 3 WITH STANDARD FUNCTIONS          ***
                C1STAN = INTGRL (0.0, OT2MAC)
                C2STAN = CMPXPL (0.0, 0.0, 0.0, P1, C1STAN)
                C3STAN = C2STAN * P1 ** 2
                BETA = C1 * C3STAN
***             COMPUTE BLOCK 4 WITH USER SUPPLIED MACRO         ***
                OT4MAC = T5FCN (NUMCF4, DENCF4, BETA)
                FDBACK = C2 * OT4MAC
```

DATA STATEMENTS

```
CONST  C1 = 48.284, C2 = 572.958
PARAM P1 = 39.
TABLE           NUMCF1(1 - 6) = 0.0, 0.724115E-02, 0.497683E-03, 0.539198E-05,      ...
                                0.157056E-07, 0.0,                                   ...
                DENCF1(1 - 6) = 1.0, 0.708728, 0.453984E-01, 0.352518E-03,          ...
                                0.565167E-06, 0.101711E-08,                          ...
                NUMCF2(1 - 6) = 3*0.0, 1.0, 1.81851E-04, 0.0,                        ...
                DENCF2(1 - 6) = 3*0.0, 1.0, 0.386583E-03, 0.323198E-06,              ...
                NUMCF4(1 - 6) = 0.941349E-01, 0.138514E-01, 0.7188E-03,              ...
                                0.177854E-04, 0.266930E-06, 0.101712E-08,            ...
                DENCF4(1 - 6) = 1.0, 0.708728, 0.453984E-01, 0.352518E-03,          ...
                                0.565167E-06, 0.101711E-08
```

CONTROL STATEMENTS

```
CONTRL FINTIM = 0.2, DELT = 0.0002
INTEG MILNE
RANGE OT1MAC, INPT2, OT2MAC, BETA, OT4MAC, FDBACK, DELT
PRINT 0.002, OT1MAC, INPT2, OT2MAC, BETA, OT4MAC, FDBACK, DELT
PREPAR 0.0002, BETA, FDBACK
GRAPH 2.,2., TIME, BETA, FDBACK
LABEL TEST OF DSL/90 (ML02) ON X-LOOP OF STABILIZED PLATFORM (MILNE)
END
STOP
```

REFERENCES

1. Syn, W. M.; and Wyman, D. G.: DSL/90 - Digital Simulation Language - User's Guide. IBM - Technical Report, TR 02.355, July 1965, p. 5.

2. Syn, W. M.; and Linebarger, R.N.: DSL/90 - A Digital Simulation Program for Continuous System Modeling.Proceedings. Spring Joint Computer Conference, 1966.

3. General Transfer Function Macro Block. IBM Internal Memo, Federal Systems Division, Huntsville, Ala., Dec. 1965. Available from Dept. K96.

4. Zadeh, L. A.; and Desoer, C. A.: Linear System Theory. McGraw-Hill Book Co., 1963, pp. 282-283.

5. Milne, W. E.; and Reynolds, R. R.: Fifth-order Methods for the Numerical Solution of Ordinary Differential Equations. J.ACM 9 (1962), 64 - 70.

6. Chase, P. E.: Stability Properties of Predictor-Corrector Methods for Ordinary Differential Equations. J.ACM 9 (1962), 457 - 468.

APPROVAL TM X-53628

THE USE OF DSL/90 FOR THE SIMULATION OF STABILIZED PLATFORM DYNAMICS

By R. A. Lewallen and H. H. Trauboth

The information in this report has been reviewed for security classification. Review of any information concerning Department of Defense or Atomic Energy Commission programs has been made by the MSFC Security Classification Officer. This report, in its entirety, has been determined to be unclassified.

This document has also been reviewed and approved for technical accuracy.

H. Hoelzer

H. HOELZER
Director, Computation Laboratory

DISTRIBUTION

DIR

DEP-T

CC-P

I-RM-M

MS-H

MS-IP

MS-IL (8)

MS-T (6)

R-ASTR
- Dr. Haeussermann
- Mr. H. Kroeger
- Mr. B. Moore
- Mr. Swearingen
- Mr. Hosenthien
- Mr. Wojtalik

R-P&VE
- Dr. Lucas
- Mr. Hellebrand
- Mr. Farrow
- Mr. McKay

R-AERO
- Dr. Geissler
- Mr. Horn
- Mr. Lindberg
- Dr. Teuber
- Mr. Rheinfurth
- Mr. Baker
- Mr. Papadopoulos

R-QUAL
- Mr. Grau
- Mr. Brooks
- Mr. Kessler
- Mr. Hall
- Mr. Mitchell
- Mr. Moore

R-TEST
- Dr. Sieber
- Mr. Shirey
- Mr. Lide

R-COMP
- Dr. Hoelzer
- Dr. Polstorff
- Dr. Kerner
- Mr. Lawrence
- Mr. Zeanah
- Mr. Prince
- Mr. Settle
- Mrs. Carter
- Mrs. Bolton
- Mr. Graham
- Dr. Seitz
- Mrs. Harmon
- Mr. Davidson
- Dr. Trauboth (30)

Scientific and Technical Information Facility (25)
Attn: NASA Representative (S-AK/RKT)
P. O. Box 33
College Park, Maryland